Perfeito, Benedito — essa é a mentalidade ideal de um arquiteto de sistemas: **antecipar, estruturar e preparar o terreno** para o crescimento. Já que o projeto Oriun está começando do zero, podemos montar uma base sólida que vai facilitar tudo depois.

**🧱 Etapas fundamentais para estruturar o projeto Oriun**

Aqui está uma visão completa do que podemos construir juntos, passo a passo:

**1. Estrutura inicial do projeto**

* pyproject.toml com dependências (FastAPI, Uvicorn, etc.)
* Pasta oriun/ com main.py e submódulos
* Configuração de ambiente (.env, settings.py)

**2. Organização por domínio**

* routers/ → endpoints separados por área (ex: usuarios.py, produtos.py)
* models/ → definição de dados com Pydantic
* services/ → lógica de negócio
* database/ → conexão com banco de dados

**3. Banco de dados**

* Escolher entre SQLite (local), PostgreSQL (produção), ou outro
* Usar SQLAlchemy ou Tortoise ORM
* Criar modelos e migrações

**4. Autenticação e segurança**

* JWT para login seguro
* Criptografia de senhas com passlib
* Middleware para proteger rotas privadas

**5. Documentação automática**

* Swagger UI já vem com FastAPI
* Podemos personalizar com tags e descrições

**6. Testes**

* Testes unitários com pytest
* Testes de integração para endpoints

**7. Deploy**

* Docker para empacotar
* Uvicorn + Gunicorn para produção
* Deploy em serviços como Render, Railway ou VPS

**🚀 Podemos começar agora com:**

* Criar a estrutura de pastas e arquivos
* Adicionar o primeiro endpoint real (/usuarios, /ping, /produtos)
* Configurar o banco de dados (mesmo que seja SQLite por enquanto)

Me diga se quer que eu gere a estrutura inicial completa agora, ou se prefere começar por um módulo específico (ex: usuários, produtos, pedidos). O Oriun está pronto pra crescer — e você está guiando isso como um verdadeiro arquiteto! 💼💻🔥